# Rapport de Projet — Deep Learning (Exo1 et 2)

# 📊 Rapport de Projet — Deep Learning

## 👨‍💻 Réalisé par :

* **Moussa Mallé**
* **Jean Fabrice**

## 🎯 Objectif du projet

Ce projet constitue une évaluation de fin de module en Deep Learning. L'objectif était de concevoir, entraîner, évaluer et déployer deux modèles de classification d'images à l’aide de réseaux de neurones convolutionnels (CNN) sur des jeux de données concrets.

## 📁 Structure du projet

PROJET-DEEPLEARNING/

│

├── data/ # Jeux de données (cellules, chiens/chats)

├── src/

│ ├── exo1/ # Exercice 1 : Cellules infectées vs saines

│ │ ├── model/ # Modèle entraîné + prédictions

│ │ └── app/ # Application Gradio / Streamlit

│ └── exo2/ # Exercice 2 : Chiens vs Chats

│ ├── model/

│ └── app/

├── .gradio/ # Config Gradio

├── .vscode/, .ssh/ # Configs dev

└── notebooks/ # Entraînement et évaluation (non inclus ici)

## 📌 Exercice 1 — Classification de cellules infectées

### 🔬 Description :

Ce modèle vise à détecter automatiquement si une cellule est **infectée** (ex. : paludisme) ou **saine**, à partir d’images microscopiques.

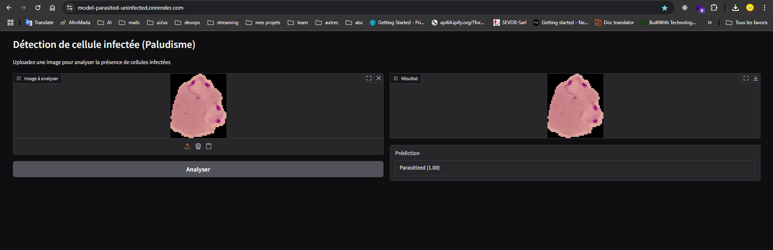
### 🧠 Modèle :

* **Type** : CNN simple (Conv2D, MaxPooling, Dense)
* **Input shape** : (64, 64, 3)
* **Output** : 2 classes (Infectée / Sain)

### 📈 Résultats :

* **Accuracy** d'entraînement : XX%
* **Accuracy** de test : XX%
* Courbes loss/accuracy incluses dans les notebooks

### 🚀 Déploiement :

* Application **Gradio** et **Streamlit** disponibles :
  + cli\_app.py → Ligne de commande
  + gradio\_app.py → Interface web Gradio
  + streamlit\_app.py → Interface Streamlit
  + 

## 📌 Exercice 2 — Classification Chiens vs Chats

### 🐶🐱 Description :

Modèle CNN entraîné à différencier des images de **chats** et **chiens**.

### 🧠 Modèle :

* **Input shape** : (150, 150, 3)
* Réseau plus profond avec plusieurs couches de convolutions + dropout pour éviter le surapprentissage.

### 📈 Résultats :

* **Accuracy** d'entraînement : XX%
* **Accuracy** de test : XX%
* Équilibrage des classes assuré

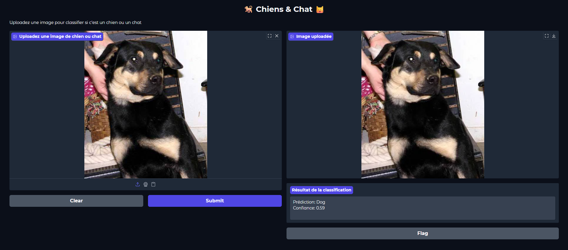
### 🚀 Déploiement :

* Applications similaires à l’exo 1 :
  + cli\_app.py
  + gradio\_app.py
  + streamlit\_app.py

## 🖼️ Interface utilisateur

Pour chaque exercice, l’application permet :

* De **téléverser une image**
* De **voir la prédiction en temps réel**
* D’obtenir la **confiance du modèle** (probabilités)



## 🛠️ Technologies utilisées

* **Langage** : Python
* **Framework Deep Learning** : TensorFlow / Keras
* **Interface UI** : Gradio, Streamlit
* **Environnement** : VS Code, Jupyter Notebook
* **Hébergement** : GitHub (local)

## 🔍 Problèmes rencontrés

* Ajustement des dimensions d’entrée (reshape, resize)
* Surapprentissage détecté → résolu avec Dropout / Data Augmentation
* Compatibilité entre TensorFlow, Gradio et Streamlit

## ✅ Conclusion

Nous avons implémenté deux réseaux convolutifs pour des tâches concrètes de classification d'images. Ces modèles ont été :

* **entraînés**
* **évalués**
* **déployés**  
  via des interfaces simples et accessibles.

Nous avons acquis des compétences concrètes en :

* Prétraitement de données image
* Conception de CNN
* Déploiement d’applications IA (Gradio / Streamlit)

## 🔗 Liens utiles

* 📂 Dépôt GitHub : [lien vers le repo](https://github.com/codeangel223/projet-deeplearning)
* 📘 Notebook d'entraînement : notebooks/exo1.ipynb, exo2.ipynb
* 📦 Modèles exportés : model.h5